Longest Common Subsequence (LCS)

Given a sequence $X = \langle x_1, x_2, \dots, x_m \rangle$, another sequence $Z = \langle z_1, z_2, \dots, z_k \rangle$ is a <u>subsequence</u> of X if there is an increasing sequence of indices $\langle i_1, i_2, \dots, i_k \rangle$ such that for all $j = 1, 2, \dots, k$, $\underline{x_{i_j} = z_j}$.

<u>Ex.</u>

$X = \langle a, b, b, a, c, b, d \rangle$

$Z = \langle b, a, b \rangle$

Given two sequences X, Y, we say Z is common subsequence of both if Z is a subsequence of both.

Ex.

$X = \langle a, b, b, a, c, b, d \rangle$

$Y = \langle a, b, a, c, d, b \rangle$

$Z = \langle a, b, a, c, d \rangle$ is a common subsequence.

In LCS, we are given X, Y. We are to find the (a) longest common subsequence in length.

Given $X = \langle x_1, x_2, \ldots, x_m \rangle$: the ith prefix of X denoted $X_i = \langle x_1, x_2, \ldots x_i \rangle$

Theorem (Optimal substructure of LCS):

Let $X = \langle x_1, x_2, \ldots x_m \rangle$ and $Y = \langle y_1, y_2, \cdots y_n \rangle$ be sequences and $Z = \langle z_1, z_2, \ldots, z_k \rangle$ be LCS of X and Y.

1. if $x_m = y_n$, then $z_k = x_m = y_n$ and Z_{k-1} is an LCS of X_{m-1}, Y_{n-1}
2. if $x_m \neq y_n$, then $z_k \neq x_m \Rightarrow Z$ is an LCS of X_{m-1} and Y.
3. if $x_m \neq y_n$, then $z_k \neq y_n \Rightarrow Z$ is an LCS of X and Y_{n-1}

$X = \langle a, a, b, b, a, c, b \rangle$
$Y = \langle a, b, a, d, c, d, b \rangle$
$Z = \langle a, b, a, c, b \rangle$ LCS $\Rightarrow \langle a, b, a, c \rangle$ LCS of
$X' = \langle a, a, b, b, a, c \rangle$
$Y' = \langle a, b, a, d, c, d \rangle$

A recursive solution:

Let $c[i,j]$ be the length of an LCS of prefixes X_i and Y_j.

$$c[i,j] = \begin{cases} 0 & \text{if } i=0 \text{ or } j=0 \\ c[i-1,j-1]+1 & \text{if } i,j>0 \text{ and } x_i = y_j \\ \max\{c[i,j-1],\, c[i-1,j]\} & \text{if } i,j>0 \text{ and } x_i \neq y_j \end{cases}$$

Algorithm:

```
LCS(X, Y)
m := Length(X) ; n := Length(Y)
for i := 1 to m do c[i, 0] := 0;
for j := 0 to n do c[0, j] := 0;
for i := 1 to m
    for j := 1 to n
        if xi = xj then
            c[i,j] := c[i-1, j-1]+1
            b[i,j] := "↖"
        else if c[i-1, j] ≥ c[i, j-1]
            then c[i,j] := c[i-1, j]
                 b[i,j] := "↑"
            else c[i,j] := c[i, j-1]
                 b[i,j] := "←"
    endfor
endfor
return b and c.
```

		y_j	b	d	c	a	b	a
		0	1	2	3	4	5	6
x_i	0	0	0	0	0	0	0	0
a	1	0	0	0	0	1	1	1
b	2	0	↖1	←1	1	1	2	2
c	3	0	1	1	↖2	←2	2	2
b	4	0	1	1	2	2	↖3	3
d	5	0	1	2	2	2	↑3	3
a	6	0	1	2	2	3	3	↖4
b	7	0	1	2	2	3	4	↑4

Matrices are filled up top to bottom and left to right.

```
Print LCS (b, X, i, j)
  if i=0 or j=0 then return
  if b[i,j] = "↖" then PrintLCS(b, X, i-1, j-1)
        print xi
  elseif b[i,j] = "↑" then PrintLCS(b, X, i-1, j)
  else PrintLCS(b, X, i, j-1)
```

Time and Space complexity ??